AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for reducing ~~the overhead~~ an overhead involved in executing native code methods in an application running on a virtual machine, comprising:

selecting a call to any native code method to be optimized within the virtual machine;

decompiling at least part of the ~~selected~~ native code method for the selected call into an intermediate representation, wherein an intermediate representation includes a set of instruction code which is not in final executable form;

obtaining an intermediate representation associated with the application running on the virtual machine which interacts with the ~~selected~~ native code method for the selected call;

integrating the intermediate representation for the ~~selected~~ native code method for the selected call into the intermediate representation associated with the application running on the virtual machine to form an integrated intermediate representation; and

generating ~~native~~ a native code from the integrated intermediate representation, wherein ~~the native code generation process optimizes~~ generating the native code from the integrated intermediate representation involves optimizing interactions between the application running on the virtual machine and the ~~selected~~ native code method for the selected call, wherein optimizing the

interactions involves optimizing calls from the application to the ~~selected~~ native code method for the selected call by using additional information from the integrated intermediate representation to reduce ~~the number~~ a number of indirect calls and indirect references associated with the calls from the application to the ~~selected~~ native code method for the selected call.

2. (Currently Amended) The method of claim 1, wherein selecting the call to any native code method involves selecting the call based upon at least one of:

~~the execution~~ an execution frequency of the call; and

~~the overhead~~ an overhead involved in performing the call ~~to the selected native code method~~ as compared against ~~the amount~~ an amount of work performed by the ~~selected~~ native code method for the call.

3 (Canceled).

4. (Currently Amended) The method of claim 1, wherein optimizing interactions between the application running on the virtual machine and the ~~selected~~ native code method for the selected call involves optimizing callbacks by the ~~selected~~ native code method for the selected call into the virtual machine.

5. (Currently Amended) The method of claim 4, wherein optimizing callbacks by the ~~selected~~ native code method for the selected call into the virtual machine involves optimizing callbacks that access heap objects within the virtual machine.

6. (Currently Amended) The method of claim 4, wherein the virtual machine is a platform-independent virtual machine; and

wherein integrating the intermediate representation for the ~~selected~~ native code method for the selected call with the intermediate representation associated with the application running on the virtual machine involves integrating calls provided by an interface for accessing native code into the ~~selected~~ native code method for the selected call.

7. (Original) The method of claim 1, wherein obtaining the intermediate representation associated with the application running on the virtual machine involves recompiling a corresponding portion of the application.

8. (Original) The method of claim 1, wherein obtaining the intermediate representation associated the application running on the virtual machine involves accessing a previously generated intermediate representation associated with the application running on the virtual machine.

9. (Currently Amended) The method of claim 1, wherein prior to decompiling the ~~selected~~ native code method for the selected call, the method further comprises setting up a context for the decompilation by:

determining a signature of the ~~call to the selected native code method; and~~ selected call; and

determining a mapping from arguments of the selected call to corresponding locations in a native application binary interface (ABI).

10. (Currently Amended) A computer-readable storage device storing instructions that when executed by a computer cause the computer to perform a method for reducing ~~the overhead~~ an overhead involved in executing native code methods in an application running on a virtual machine, the method comprising:

selecting a call to any native code method to be optimized within the virtual machine;

decompiling at least part of the ~~selected~~ native code method for the selected call into an intermediate representation, wherein an intermediate representation includes a set of instruction code which is not in final executable form;

obtaining an intermediate representation associated with the application running on the virtual machine which interacts with the ~~selected~~ native code method for the selected call;

integrating the intermediate representation for the ~~selected~~ native code method for the selected call into the intermediate representation associated with the application running on the virtual machine to form an integrated intermediate representation; and

generating ~~native~~ a native code from the integrated intermediate representation, wherein ~~the native code generation process optimizes~~ generating the native code from the integrated intermediate representation involves optimizing interactions between the application running on the virtual machine and the ~~selected~~ native code method for the selected call, wherein optimizing the interactions involves optimizing calls from the application to the ~~selected~~ native code method for the selected call by using additional information from the integrated intermediate representation to reduce ~~the number~~ a number of indirect calls and indirect references associated with the calls from the application to the ~~selected~~ native code method for the selected call.

11. (Currently Amended) The computer-readable storage device of claim 10, wherein selecting the call to any native code method involves selecting the call based upon at least one of:

~~the execution~~ an execution frequency of the call; and

~~the overhead~~ an overhead involved in performing the call ~~to the selected native code method~~ as compared against ~~the amount~~ an amount of work performed by the ~~selected~~ native code method for the call.

12 (Canceled).

13. (Currently Amended) The computer-readable storage device of claim 10, wherein optimizing interactions between the application running on the virtual machine and the ~~selected~~ native code method for the selected call involves optimizing callbacks by the ~~selected~~ native code method for the selected call into the virtual machine.

14. (Currently Amended) The computer-readable storage device of claim 13, wherein optimizing callbacks by the ~~selected~~ native code method for the selected call into the virtual machine involves optimizing callbacks that access heap objects within the virtual machine.

15. (Currently Amended) The computer-readable storage device of claim 13,

wherein the virtual machine is a platform-independent virtual machine; and

wherein ~~integrting~~ integrating the intermediate representation for the ~~selected~~ native code method for the selected call with the intermediate representation associated with the application running on the virtual machine involves integrating calls provided by an interface for accessing native code into the ~~selected~~ native code method for the selected call.

16. (Previously presented) The computer-readable storage device of claim 10, wherein obtaining the intermediate representation associated with the application running on the virtual machine involves recompiling a corresponding portion of the application.

17. (Previously presented) The computer-readable storage device of claim 10, wherein obtaining the intermediate representation associated with the application running on the virtual machine involves accessing a previously generated intermediate representation associated with the application running on the virtual machine.

18. (Currently Amended) The computer-readable storage device of claim 10, wherein prior to decompiling the ~~selected~~ native code method for the selected call, the method further comprises setting up a context for the decompilation by:

determining a signature of the ~~call to the selected native code method; and~~selected call; and

determining a mapping from arguments of the selected call to corresponding locations in a native application binary interface (ABI).

19-27. (Cancelled)

28. (Currently Amended) A method for reducing ~~the overhead~~ an overhead involved in executing native code methods in an application running on a virtual machine, comprising:

deciding to optimize a callback by any native code method into the virtual machine;

decompiling at least part of the ~~selected~~ native code method for the callback into an intermediate representation, wherein an intermediate representation includes a set of instruction code which is not in final executable form;

obtaining an intermediate representation associated with the application running on the virtual machine which interacts with the ~~selected~~ native code method for the callback;

integrating the intermediate representation for the ~~selected~~ native code method for the callback into the intermediate representation associated with the application running on the virtual machine to form an integrated intermediate representation; and

generating ~~native~~ a native code from the integrated intermediate representation, wherein ~~the native code generation process optimizes the callback by any native code method into the virtual machine,~~ generating the native code from the integrated intermediate representation involves optimizing the callback, wherein optimizing the callback involves optimizing calls from the ~~selected~~ native code method for the callback to the application by using additional information from the integrated intermediate representation to reduce ~~the number~~ a number of indirect calls and indirect references associated with the calls from the ~~selected~~ native code method for the callback to the application.

29. (Currently Amended) The method of claim 28, wherein ~~the native code generation process also optimizes calls to the selected native code method by the application.~~ generating the native code from the integrated intermediate representation also involves optimizing calls by the application to the native code method for the callback.

30. (Previously Presented) The method of claim 28, wherein optimizing the callback by any native code method into the virtual machine involves optimizing a callback that accesses a heap object within the virtual machine.

31. (Currently Amended) The method of claim 28,

wherein the virtual machine is a platform-independent virtual machine; and

wherein integrating the intermediate representation for the ~~selected~~ native code method for the callback with the intermediate representation associated with the application running on the virtual machine involves integrating calls provided by an interface for accessing native code into the ~~selected~~ native code method for the callback.

32. (Currently amended) A computer-readable storage device storing instructions that when executed by a computer cause the computer to perform a method for reducing ~~the overhead~~ an overhead involved in executing native code methods in an application running on a virtual machine, the method comprising:

deciding to optimize a callback by any native code method into the virtual machine;

decompiling at least part of the ~~selected~~ native code method for the callback into an intermediate representation, wherein an intermediate representation includes a set of instruction code which is not in final executable form;

obtaining an intermediate representation associated with the application running on the virtual machine which interacts with the ~~selected~~ native code method for the callback;

integrating the intermediate representation for the ~~selected~~ native code method for the callback into the intermediate representation associated with the application running on the virtual machine to form an integrated intermediate representation; and

generating ~~native~~ a native code from the ~~combined~~ integrated intermediate representation, wherein ~~the native code generation process optimizes the callback by any native code method into the virtual machine,~~ generating the native code from the integrated intermediate representation involves optimizing the callback, wherein optimizing the callback involves optimizing calls from the ~~selected~~ native code method for the callback to the application by using additional information from the integrated intermediate representation to reduce ~~the number~~ a number of indirect calls and indirect references associated with the calls from the ~~selected~~ native code method for the callback to the application.

33. (Currently Amended) The computer-readable storage device of claim 32, wherein ~~the native code generation process also optimizes calls to the selected native code method by the application.~~ generating the native code from the integrated intermediate representation also involves optimizing calls by the application to the native code method for the callback.

34. (Previously Presented) The computer-readable storage device of claim 32, wherein optimizing the callback by any native code method into the virtual machine involves optimizing a callback that accesses a heap object within the virtual machine.

35. (Currently Amended) The computer-readable storage device of claim 32, wherein the virtual machine is a platform-independent virtual machine; and

wherein integrating the intermediate representation for the ~~selected~~ native code method for the callback with the intermediate representation associated with the application running on the virtual machine involves integrating calls provided by an interface for accessing native code into the ~~selected~~ native code method for the callback.

36-39. (Canceled)